UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

COMPUTACIÓN GRÁFICA e INTERACCIÓN HUMANO COMPUTADORA

TECHNICAL MANUAL

CLASS: 05

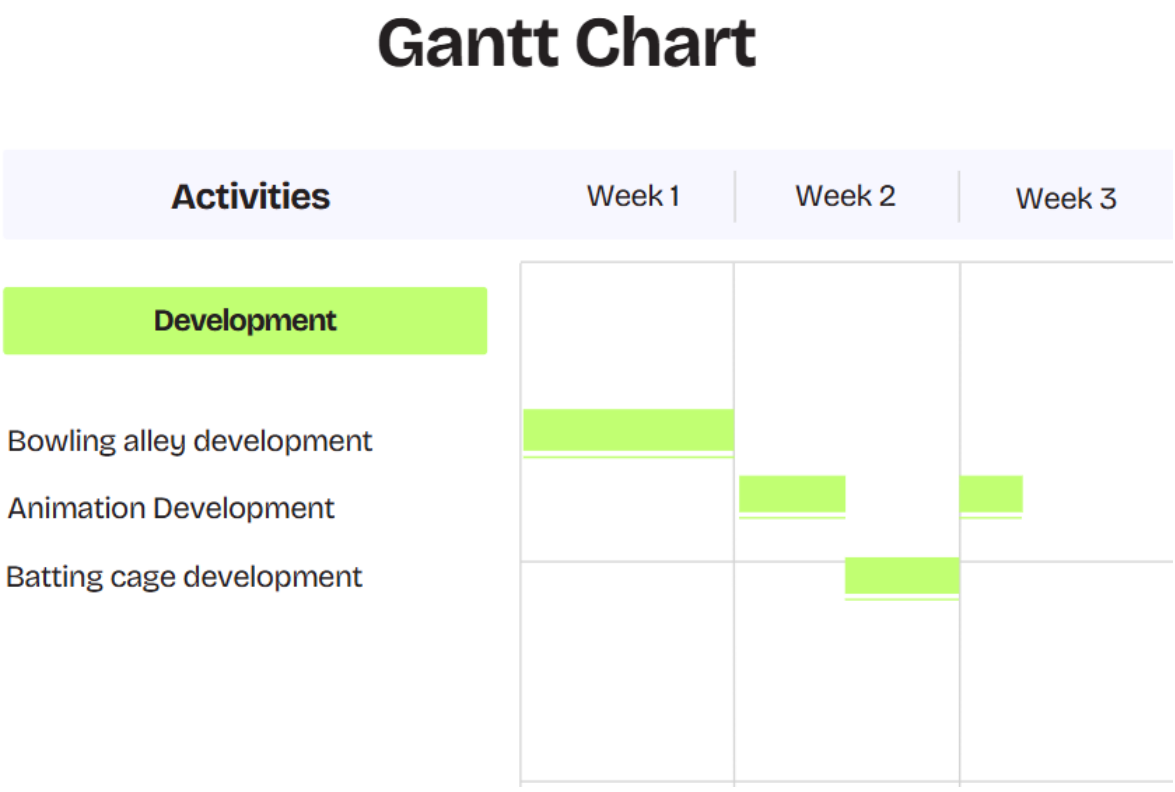
SEMESTER 2025-2

Objective

Design, develop, and implement two attractions to be part of an interactive fair, with an environment and dynamics inspired by the video game *Ultrakill*. These attractions consist of a bowling alley and a batting cage. They will showcase the setting and elements from the "Ship of Fools" level and the "Cerberus" level, respectively.

Project scope

- Diseño de ambas atracciones basado en la estética y mecánicas de *Ultrakill*.
- Design both attractions based on Ultrakill's aesthetic and mechanics.
- Bowling alley and batting cage implementation that includes thematic decorations
- Develop animations to be executed when interacting with the attractions



Function and Variable Dictionary:

Bowling Alley

- void renderBuilding: Applies transformations and renders the meshes to build the bowling alley structure.
- void renderPines: Applies transformations and renders the models for the bowling pins.
- void renderPinesAnimation: Applies transformations and renders the bowling pins models; also includes transformations with variables to enable animations.

- void renderTerminal: Applies transformations and renders the models for the bowling terminal.
- void renderMauriceBowling: Applies transformations and renders the bowling ball model.
- void renderTable: Applies transformations and renders the models for the bowling tables.
- void renderChandelier: Applies transformations and renders the chandelier models.
- void renderLaneFloor: Applies transformations and renders the mesh for the bowling lane floors.
- void renderRailing: Applies transformations and renders the models for the lane railings.
- void renderCarpet: Applies transformations and renders the carpet models.
- void renderChair: Applies transformations and renders the models for the chairs at each table in the bowling alley.
- void renderCerberusStatue: Applies transformations and renders the decorative Cerberus statues.
- void renderBoliche: Function that renders the bowling alley elements.
    - glm::mat4 model: Matrix variable that stores the transformations.
    - GLuint uniformModel: Integer variable for the uniform model.
    - std::vector<Model*> listaModelos: Vector containing pointers to the model objects used in the bowling alley.
    - std::vector<Mesh*> bowlingMeshList: Vector storing the meshes created for the bowling alley (e.g., floor, lanes, walls, ceiling).
    - std::vector<Texture*> listaTexturas: Vector with pointers to the textures used by the meshes.
    - GLboolean ePressed: Boolean indicating whether the E key is pressed.
    - glm::vec3 modelPosition: Variable containing the avatar model's position.
    - GLfloat* bowlingAnimationLane: Pointer to an array of floats, each position represents transformation variability for different bowling lanes.
    - GLfloat deltaTime: Difference between the current and previous frame, used to standardize animation speed across framerates.
    - GLfloat altura: Height at which the bowling alley models are positioned.
    - GLfloat z: Stores the Z-position of the elements; reduced in a for loop to create multiple instances of a bowling lane.

Batting Cage

- void renderBattingBuilding: Applies transformations and renders the meshes for the building containing the batting cages.
- void renderFeedbacker: Applies transformations and renders the bat.
- void renderOrbBall: Applies transformations and renders the ball.
- void renderLava: Applies transformations and renders the lava floor mesh.
- void renderCerberusPitcher: Applies transformations and renders the pitcher model.
- void renderReja: Applies transformations and renders the batting cage fences.
- void renderV1: Applies transformations and renders the V1 model.
- void renderBatting: Groups the rendering functions of the batting cage elements and also applies animations.
    - glm::mat4 model: Matrix variable that stores transformations.
    - GLuint uniformModel: Integer variable for the uniform model.
    - std::vector<Model*> listaModelos: Vector containing pointers to the model objects used in the batting cage.
    - std::vector<Mesh*> MeshList: Vector containing pointers to the meshes for floor, walls, ceiling, and lava.
    - std::vector<Texture*> listaTexturas: Vector with pointers to the textures used by the meshes.
    - GLfloat* battingAnimationLane: Pointer to an array with values used to modify transformation magnitudes for animations.
    - GLfloat deltaTime: Frame time difference used to standardize animation speed across framerates.
    - GLboolean ePressed: Boolean indicating whether the E key is pressed.
    - glm::vec3 modelPosition: Variable containing the avatar model's position.
    - GLboolean* battingReverse: Boolean used to reverse the ball's movement direction when hit.
    - GLfloat* parryAnimation: Pointer to a float array, where each position represents transformation variability for different batting cages.
    - GLboolean* parryReverse: Boolean indicating whether the bat moves in the opposite direction after hitting.
    - GLfloat z: Stores the Z-position of the elements; reduced in a for loop to create multiple instances of a bowling lane.

Cost

Below is a breakdown of activities with the invested hours.

DATE	ACTIVITIES	HOURS
15/04/2025	The repository and the initial files were uploaded, deprecated files were removed	1
16/04/2025	Ultrakill models were added, and variation to the sun intensity was added. Floor and grass texture was added. The avatar was added, with positioning simulating a third person camera	6
17/04/2025	Added skybox variation for the day night cycle. Added more models, mesh and textures for the bowling alley. Added a teleport function	10
18/04/2025	Moved bowling alley elements underground. Added teleporting into and out of the bowling alley	3
19/04/2025	Added railing and floor to each lane	3
20/04/2025	Added last models for bowling alley, esquites stall, and angle variation for the sun light	5
24/04/2025	The avatar's limbs were separated and implemented with hierarchy. A walking animation was implemented	4

25/04/2025	Walking animation upgrades	2
28/04/2025	Batting cage structure was added, along with V1 and V2 NPC's	3
30/04/2025	Added cameras, along with interaction with the terminals. Added bowling animation	6
07/05/2025	Added batting cage interaction with the terminals to activate the animation. Fixed sun's x component. An arc with an animated sign	4

In total, 47 hours were invested into the project's development.

Taking an hourly rate of MXN$200.00, the total labor cost is MXN$9400.00.

Depreciation costs of the hardware is also taken into account, below is the breakdown

	Cost	Lifespan (Hours)	Depreciation cost per hour	Total
PC	$18,000	60480	$0.30	$13.99
Monitor	$4000	60480	$0.07	$3.11
Monitor	$10000	60480	$0.17	$7.77
Teclado	$800	30240	$0.03	$1.24
Mouse	$800	30240	$0.03	$1.24
Internet	$900	840	$1.07	$50.36

Resulting in a total of MXN$77.71.
The cost of electrical energy is also added. Considering a consumption of 0.2 kWh and a cost of MXN$2.00 per kWh, the total is MXN$18.80.
A subtotal of MXN$9,496.51 is obtained, to which a 30% profit margin is added, resulting in a cost

of MXN$12,345.47.
Finally, 16% VAT is added, giving a grand total of MXN$14,320.74.

Conclusions

Upon completing the development of the project, the objective of creating two interactive fair attractions themed around *Ultrakill*, integrated into a three-dimensional digital environment, was achieved. These attractions allow the user to explore the space and interact with mechanics inspired by the game, such as parrying and activating elements within the scene. The result offers an interactive experience that combines visual references from the game with an environment designed to encourage exploration.